UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/266,675	03/11/1999	RANDY S. KIMMERLY	777.278US1	6126

7590 03/25/2004

Steven J Rocci
Woodcock Washburn Kurtz MacKiewicz & Norris LLP
One Liberty Place
46th Floor
Philadelphia, PA 19103

EXAMINER	
LY, ANH	

ART UNIT	PAPER NUMBER
2172	20

DATE MAILED: 03/25/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/266,675	KIMMERLY, RANDY S.	
	Examiner	Art Unit	
	Anh Ly	2172	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *03 March 2004*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-24* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-24* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

Request for Continued Examination

1. The request filed on 03/03/2004 for a Request for Continued Examination (RCE) under 37 CFR 1.114 based on parent Application No. 09/266,675 is acceptable and a RCE has been established. An action on the RCE follows.

2. Claims 1-24 are pending in this application.

Claim Rejections - 35 USC § 103

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. Claims 1-3, 5-6, 10-18, and 22-23 are rejected under 35 U.S.C. 103(a) as being unpatentable over US Patent No 5,187,786 issued to Densmore et al. (hereinafter Densmore) in view of US Patent No. 6,609,146 issued to Slotznick.

With respect to claim 1, Densmore discloses generating a cache of information relating to the classes in the class path (the implementing a root class with a class hierarchy of objects being a class path containing a plurality of class path directory names, and its elements are class files and class instance, since a path or directory is a hierarchical structure: abstract, col. 2, lines 46-67 and col. 3, lines 1-14col. 4, lines 46-67 and col. 5, lines 1-28 and the root class and class instance directory are stored in the system: col. 4, lines 64-67); creating a wrapper (hierarchy of root class and class instance directory are a wrapper: col. col. 4, lines 60-67 and col. 5, lines 1-28);

requesting a search of the class path and searching the cache to satisfy the requested search (search path: col. 9, lines 6-9 and lines 15-32).

Densmore discloses creating a root class with a class hierarchy of objects in a hierarchical file system, objects are organized as class instances and classes, the data are contained in the class variables and/or the class instance variables, a hierarchy of a root class directory and root class files, a class making procedure comprising an interface for receiving a class specification as well as for receiving a message (abstract, col. 2, lines 46-67, col. 3, lines 1-14, col. 4, lines 46-67 and col. 5, lines 1-28), setting the search path of hierarchical file system to the content of the class instance or class file under unchanged current directory (col. 9, lines 14-22). Densmore does not explicitly a level of indirection from application programming interfaces used by a class locator, the wrapper indirection level providing for different caches to be used for the selected elements.

However, Slotznick teaches a programmable processor or CPU communicating with a various kinds of memory including a particular a fast memory cache (col. 12, lines 32-35) and a wrapper creating the search path via the call of API to search the class path (col. 20, lines 11-22).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to employ the teachings of Densmore with the teachings of Slotznick so as to have an interface for searching the class path via the Window APP from which calls the search path application stored in the computer (col. 20, lines 11-22). This combination would have made the method for generating and storing a root class or a hierarchy of root class and class instance directory and stored in the computer system (Densmore – col. 4, lines 46-67 and col. 5, lines 1-28) and searching a path or class path via a application interface (Slotznick – col.20, lines 11-22) in order to optimize the runtime for looking the class in the path or directory in the Java class file environment.

Claim 2 is essentially the same as claim 1 except that it is directed to a computer readable medium rather than a method, and is rejected for the same reason as applied to the claim 1 hereinabove.

With respect to claim 3, Densmore wherein the class path comprises multiple elements, each element having multiple classes stored therein (class hierarchy of object: comprising root class, a plurality of classes: see abstract).

With respect to claim 5, Densmore discloses generating a search request for desired classes within the multi-element class path; and independently satisfying the

request in association with each element in the class path (abstract, col. 2, lines 46-67 and col. 3, lines 1-13; a wrapper (hierarchy of root class and class instance directory are a wrapper: col. col. 4, lines 60-67 and col. 5, lines 1-28); and search path: col. 9, lines 6-9 and lines 15-32).

Densmore discloses creating a root class with a class hierarchy of objects in a hierarchical file system, objects are organized as class instances and classes, the data are contained in the class variables and/or the class instance variables, a hierarchy of a root class directory and root class files, a class making procedure comprising an interface for receiving a class specification as well as for receiving a message (abstract, col. 2, lines 46-67, col. 3, lines 1-14, col. 4, lines 46-67 and col. 5, lines 1-28), setting the search path of hierarchical file system to the content of the class instance or class file under unchanged current directory (col. 9, lines 14-22). Densmore does not explicitly a level of indirection to search the appropriate class path for the search request, two of the elements have at least two separate caches for the at least two elements.

However, Slotznick teaches a programmable processor or CPU communicating with a various kinds of memory including a particular a fast memory cache (col. 12, lines 32-35) and a wrapper creating the search path via the call of API to search the class path (col. 20, lines 11-22).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to employ the teachings of Densmore with the teachings of Slotznick so as to have an interface for searching the class path via the Window APP from which calls the search path application stored in the computer (col.

20, lines 11-22). This combination would have made the method for generating and storing a root class or a hierarchy of root class and class instance directory and stored in the computer system (Densmore – col. 4, lines 46-67 and col. 5, lines 1-28) and searching a path or class path via a application interface (Slotznick – col.20, lines 11-22) in order to optimize the runtime for looking the class in the path or directory in the Java class file environment.

Claim 6 is essentially the same as claim 5 except that it is directed to a computer readable medium rather than a method, and is rejected for the same reason as applied to the claim 5 hereinabove.

With respect to claim 10, Densmore discloses parsing the class path into names of elements (col. 7, lines 65-67 and col. 8, lines 1-26); determining which elements are viable for caching; and initiating creation of caches and wrappers (hierarchy of root class and class instance directory are a wrapper: col. col. 4, lines 60-67 and col. 5, lines 1-28) for those elements which are viable (see abstract, see figs: 1, 3 and 4, col. 5, lines 54-67 and col. 6, lines 1-45 and col. 7, lines 55-67 and col. 8, lines 1-26).

Densmore discloses creating a root class with a class hierarchy of objects in a hierarchical file system, objects are organized as class instances and classes, the data are contained in the class variables and/or the class instance variables, a hierarchy of a root class directory and root class files, a class making procedure comprising an interface for receiving a class specification as well as for receiving a message (abstract, col. 2, lines 46-67, col. 3, lines 1-14, col. 4, lines 46-67 and col. 5, lines 1-28), setting the search path of hierarchical file system to the content of the class instance or class

file under unchanged current directory (col. 9, lines 14-22). Densmore does not explicitly a level of indirection to search the appropriate class path for the search request, two of the elements have at least two separate caches for the at least two elements.

However, Slotznick teaches a programmable processor or CPU communicating with a various kinds of memory including a particular a fast memory cache (col. 12, lines 32-35) and a wrapper creating the search path via the call of API to search the class path (col. 20, lines 11-22).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to employ the teachings of Densmore with the teachings of Slotznick so as to have an interface for searching the class path via the Window APP from which calls the search path application stored in the computer (col. 20, lines 11-22). This combination would have made the method for generating and storing a root class or a hierarchy of root class and class instance directory and stored in the computer system (Densmore – col. 4, lines 46-67 and col. 5, lines 1-28) and searching a path or class path via a application interface (Slotznick – col.20, lines 11-22) in order to optimize the runtime for looking the class in the path or directory in the Java class file environment.

With respect to claims 11-14, Densmore discloses wherein the viability of an element for caching is dependent on the ease of tracking which elements have had changes in them; wherein the viability of an element for caching is determined based on it being a predetermined type; checking a registry to see if the element already has a

cache associated with it; and determining if an existing cache is up to date (col. 5, lines 54-67, col. 6, lines 1-67, col. 7, lines 1-67 and col. 8, lines 1-26).

With respect to claim 15, Densmore discloses means for receiving requests to search a multi-elements class path for classes (search path: col. 9, lines 6-9 and lines 15-32), means for transferring such request through a wrapper associated (hierarchy of root class and class instance directory are a wrapper: col. col. 4, lines 60-67 and col. 5, lines 1-28).

Densmore discloses creating a root class with a class hierarchy of objects in a hierarchical file system, objects are organized as class instances and classes, the data are contained in the class variables and/or the class instance variables, a hierarchy of a root class directory and root class files, a class making procedure comprising an interface for receiving a class specification as well as for receiving a message (abstract, col. 2, lines 46-67, col. 3, lines 1-14, col. 4, lines 46-67 and col. 5, lines 1-28), setting the search path of hierarchical file system to the content of the class instance or class file under unchanged current directory (col. 9, lines 14-22). Densmore does not explicitly a level of indirection to search the appropriate class path for the search request, two of the elements have at least two separate caches for the at least two elements.

However, Slotznick teaches a programmable processor or CPU communicating with a various kinds of memory including a particular a fast memory cache (col. 12, lines 32-35) and a wrapper creating the search path via the call of API to search the class path (col. 20, lines 11-22).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to employ the teachings of Densmore with the teachings of Slotznick so as to have an interface for searching the class path via the Window APP from which calls the search path application stored in the computer (col. 20, lines 11-22). This combination would have made the method for generating and storing a root class or a hierarchy of root class and class instance directory and stored in the computer system (Densmore – col. 4, lines 46-67 and col. 5, lines 1-28) and searching a path or class path via a application interface (Slotznick – col.20, lines 11-22) in order to optimize the runtime for looking the class in the path or directory in the Java class file environment.

With respect to claim 16, Densmore disclose at least one such element specific search method comprising a cache associated with such element (col. 9, lines 1-32 and col. 4, lines 60-67)

With respect to claim 17, Densmore discloses means for parsing the multi-elements class path into names of elements (col. 7, lines 65-67 and col. 8, lines 1-26); means for determining whether each element is a variable cache candidate and for creating a cache for such variable candidates (col. 5, lines 54-67, col. 6, lines 1-67, col. 7, lines 1-67 and col. 8, lines 1-26) and means for creating indirection wrappers (hierarchy of root class and class instance directory are a wrapper: col. col. 4, lines 60-67 and col. 5, lines 1-28).

Densmore discloses creating a root class with a class hierarchy of objects in a hierarchical file system, objects are organized as class instances and classes, the data

are contained in the class variables and/or the class instance variables, a hierarchy of a root class directory and root class files, a class making procedure comprising an interface for receiving a class specification as well as for receiving a message (abstract, col. 2, lines 46-67, col. 3, lines 1-14, col. 4, lines 46-67 and col. 5, lines 1-28), setting the search path of hierarchical file system to the content of the class instance or class file under unchanged current directory (col. 9, lines 14-22). Densmore does not explicitly a level of indirection to search the appropriate class path for the search request, two of the elements have at least two separate caches for the at least two elements.

However, Slotznick teaches a programmable processor or CPU communicating with a various kinds of memory including a particular a fast memory cache (col. 12, lines 32-35) and a wrapper creating the search path via the call of API to search the class path (col. 20, lines 11-22).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to employ the teachings of Densmore with the teachings of Slotznick so as to have an interface for searching the class path via the Window APP from which calls the search path application stored in the computer (col. 20, lines 11-22). This combination would have made the method for generating and storing a root class or a hierarchy of root class and class instance directory and stored in the computer system (Densmore – col. 4, lines 46-67 and col. 5, lines 1-28) and searching a path or class path via a application interface (Slotznick – col.20, lines 11-22) in order to optimize the runtime for looking the class in the path or directory in the Java class file environment.

With respect to claim 18, Densmore discloses the cache for each viable candidate comprises a name of class (col. 5, lines 9-15 and lines 49-60).

With respect to claim 22, Densmore discloses a class path manager that receives requests for identification or enumeration of classes of classes in the class path; a cache for a cache viable element of the class path; a wrapper for such cache viable element that receives such request from the class path manager (abstract, col. 2, lines 46-67 and col. 3, lines 1-13; also see col. 5, lines 1-42; col. 7, lines 11-18 and col. 8, lines 6-15).

Densmore discloses creating a root class with a class hierarchy of objects in a hierarchical file system, objects are organized as class instances and classes, the data are contained in the class variables and/or the class instance variables, a hierarchy of a root class directory and root class files, a class making procedure comprising an interface for receiving a class specification as well as for receiving a message (abstract, col. 2, lines 46-67, col. 3, lines 1-14, col. 4, lines 46-67 and col. 5, lines 1-28), setting the search path of hierarchical file system to the content of the class instance or class file under unchanged current directory (col. 9, lines 14-22). Densmore does not explicitly a level of indirection to search the appropriate class path for the search request, two of the elements have at least two separate caches for the at least two elements.

However, Slotznick teaches a programmable processor or CPU communicating with a various kinds of memory including a particular a fast memory cache (col. 12, lines 32-35) and a wrapper creating the search path via the call of API to search the class path (col. 20, lines 11-22).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to employ the teachings of Densmore with the teachings of Slotznick so as to have an interface for searching the class path via the Window APP from which calls the search path application stored in the computer (col. 20, lines 11-22). This combination would have made the method for generating and storing a root class or a hierarchy of root class and class instance directory and stored in the computer system (Densmore – col. 4, lines 46-67 and col. 5, lines 1-28) and searching a path or class path via a application interface (Slotznick – col.20, lines 11-22) in order to optimize the runtime for looking the class in the path or directory in the Java class file environment.

Claim 23 is essentially the same as claim 5 except that it is directed to a computer readable medium rather than a method, and is rejected for the same reason as applied to the claim 5 hereinabove.

5. Claims 4, 7-9 and 19-21 are rejected under 35 U.S.C. 103(a) as being unpatentable over US Patent No 5,187,786 issued to Densmore et al. (hereinafter Densmore) in view of US Patent No. 6,609,146 issued to Slotznick and further in view of US Patent No. 6,321,261 issued to Glass.

With respect to claim 4, Densmore in view of Slotznick discloses a method of locating classes as discussed in claim 1.

Densmore in view of Slotznick does not explicitly indicate, "a zip file."

However, Glass discloses searching directories for zip file in the class path (col. 3, lines 50-60).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to employ the teachings of Densmore with the teachings of Slotznick so as to have an interface for searching the class path (col. 7, lines 14-56). This combination would have made the method for generating and storing a root class or a hierarchy of root class and class instance directory and stored in the computer system (Densmore – col. 4, lines 46-67 and col. 5, lines 1-28) and searching a path or class path via a application interface (Wong – col. 5, lines 15-25) in order to optimize the runtime for looking the class in the path or directory (Brown - col. 2, lines 60-67) in the Java class file environment.

With respect to claims 7-9, Densmore in view of Wong discloses a method of locating classes as discussed in claim 5. And also Wong discloses packages of Java class (col. 6, lines 15-20).

Densmore in view of Wong does not explicitly indicate, "a zip file."

However, Glass discloses searching directories for zip file in the class path (col. 3, lines 50-60).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to employ the teachings of Densmore with the teachings of Wong so as to have an interface for searching the class path (col. 7, lines 14-56). This combination would have made the method for generating and storing a root class or a hierarchy of root class and class instance directory and stored in the computer system (Densmore – col. 4, lines 46-67 and col. 5, lines 1-28) and searching a path or class path via a application interface (Slotznick – col.20, lines 11-22) in order to optimize the runtime for looking the class in the path or directory (Brown - col. 2, lines 60-67) in the Java class file environment.

With respect to claim 19, Densmore in view of Slotznick discloses a method of locating classes as discussed in claim 5. And also Slotznick discloses packages of Java class (col. 6, lines 15-20).

Densmore in view of Slotznick does not explicitly indicate, "a zip file."

However, Glass discloses searching directories for zip file in the class path (col. 3, lines 50-60).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to employ the teachings of Densmore with the teachings of Slotznick so as to have an interface for searching the class path (col. 7, lines 14-56). This combination would have made the method for generating and storing a root class or a hierarchy of root class and class instance directory and stored in the

computer system (Densmore – col. 4, lines 46-67 and col. 5, lines 1-28) and searching a path or class path via a application interface (Slotznick – col.20, lines 11-22) in order to optimize the runtime for looking the class in the path or directory in the Java class file environment.

With respect to claims 20-21, Densmore discloses the directories are not caches and wherein the viability of an element for caching is dependent on the ease of tracking which elements have had changes in them (see abstract, see figs. 1, 3-4; col. 2, lines 46-65 and col. 5, lines 17-42).

6. Claim 24 is are rejected under 35 U.S.C. 103(a) as being unpatentable over US Patent No 5,187,786 issued to Densmore et al. (hereinafter Densmore) in view of US Patent No. 6,609,146 issued to Slotznick and further in view of US Patent No. 6,212,564 issued to Harter et al. (hereinafter Harter). .

With respect to claim 24, Densmore in view of Slotznick discloses a method of locating classes as discussed in claim 23.

Densmore in view of Slotznick does not explicitly indicate, "checking a data/time stamp on the element."

However, Harter discloses the current data/time storing in the cache (col. 3, lines 38-51 and col. 5, lines 1-15).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to employ the teachings of Densmore in view of Slotznick with the teachings of Harter so as to have date/time indication for cache (col. 5, lines 1-15). This combination would have made the method for generating and storing a root class or a hierarchy of root class and class instance directory and stored in the computer system (Densmore – col. 4, lines 46-67 and col. 5, lines 1-28) and searching a path or class path via a application interface (Slotznick – col.20, lines 11-22) and to have Java runtime stored in the system (Harter - col. 2, lines 20-25) in order to optimize the runtime for looking the class in the path or directory in the Java class file environment.

Contact Information

7. Any inquiry concerning this communication should be directed to An Ly whose telephone number is (703) 306-4527 via E-Mail: **ANH.LY@USPTO.GOV**. The examiner can be reached on Monday - Friday from 8:00 AM to 4:00 PM.

If attempts to reach the examiner are unsuccessful, see the examiner's supervisor, John Breene, can be reached on (703) 305-9790.

Any response to this action should be mailed to:

Commissioner of Patents and Trademarks

Washington, D.C. 20231

or faxed to: (703) 872-9306 (Central Official Fax Number)

Hand-delivered responses should be brought to Crystal Park II, 2121 Crystal Drive, Arlington, VA, Fourth Floor (receptionist).

Inquiries of a general nature or relating to the status of this application should be directed to the Group receptionist whose telephone number is (703) 305-3900.

JEAN M. CORRIELUS
PRIMARY EXAMINER

AL
MAR. 17th, 2004